CSCI 4203 Lab Assignment 1 :

Problem 1 : 10-bit ripple carry adder (50 points):

Part A (10 points) :

An ALU (arithmetic and logic unit) is a functional unit used by the execution stage of modern pipelined processors. It can do all kinds of arithmetic operations such as addition, multiplication, division, shift, logical operations and so on. Figure B.5.6 (on page B-29) of the textbook shows a 1-bit ALU, which performs AND, OR and addition. A control input selects the output of the ALU. Input 0 corresponds to AND, input 1 corresponds to OR and input 2 corresponds to addition. Implement a behavioral model of this 1-bit ALU using the given module skeleton (`one_bit_ALU.v`).

Part B (40 points) :

An n-bit adder adds two n-bit numbers and generates an n-bit result and a carry result.
An n-bit adder can be constructed by chaining n 1-bit ALUs together.
The carry-out output of each 1-bit ALU is connected to the carry-in input of the next ALU. The carry out of the last ALU in the chain is the carry result of the addition.
Implement an 10-bit ripple carry adder using ten 1-bit ALU units.
The control input of the ALU should be hard-coded to '2' to select addition.
You are provided a skeleton file for the 10-bit adder (`ripple_carry_ten_bit.v`), which should be used as a starting point.

Required Reading : Section "A 1-bit ALU" in Appendix B.5 has a description of the 1-bit ALU's.

Problem 2 : 10-bit Carry-Lookahead adder (50 points):

A carry lookahead adder is an optimized n-bit full adder implementation which achieves faster addition than an n-bit ripple carry adder by using multiple 'levels of abstraction'. The simplest carry look ahead adder uses a single layer of abstraction, where it computes *propagate* and *generate* values for each bit position and uses these values to compute carry-outs at each bit position. Even faster addition speeds can be achieved using multiple layers of abstraction.

Implement a 10-bit carry lookahead adder using a single layer of abstraction, using the given 12 module skeletons. 10 modules implement the carry-out calculation, one module implements the propagate and generate bit calculation, and one module is the top level module.
The equations for propagate, generate and carry-out for each bit position are printed below for your convenience :

gi = ai.bi (AND operation)
pi = ai + bi (XOR operation)

c1 = g0 + (p0.c0)
c2 = g1 + (p1.g0) + (p1.p0.c0)
c3 = g2 + (p2.g1) + (p2.p1.g0) + (p2.p1.p0.c0)
c4 = g3 + (p3.g2) + (p3.p2.g1) + (p3.p2.p1.g0) + (p3.p2.p1.p0.c0)
c5 = g4 + (p4.g3)+ (p4.p3.g2) + (p4.p3.p2.g1) + (p4.p3.p2.p1.g0) + (p4.p3.p2.p1.p0.c0)
c6 = g5 + (p5.g4) + (p5.p4.g3)+ (p5.p4.p3.g2) + (p5.p4.p3.p2.g1) + (p5.p4.p3.p2.p1.g0) + (p5.p4.p3.p2.p1.p0.c0)
c7 = g6 + (p6.g5) + (p6.p5.g4) + (p6.p5.p4.g3)+ (p6.p5.p4.p3.g2) + (p6.p5.p4.p3.p2.g1) + (p6.p5.p4.p3.p2.p1.g0) + (p6.p5.p4.p3.p2.p1.p0.c0)
c8 = g7 + (p7.g6) + (p7.p6.g5) + (p7.p6.p5.g4) + (p7.p6.p5.p4.g3)+ (p7.p6.p5.p4.p3.g2) + (p7.p6.p5.p4.p3.p2.g1) + (p7.p6.p5.p4.p3.p2.p1.g0) + (p7.p6.p5.p4.p3.p2.p1.p0.c0)
c9 = g8 + (p8.g7) + (p8.p7.g6) + (p8.p7.p6.g5) + (p8.p7.p6.p5.g4) + (p8.p7.p6.p5.p4.g3)+ (p8.p7.p6.p5.p4.p3.g2) + (p8.p7.p6.p5.p4.p3.p2.g1) + (p8.p7.p6.p5.p4.p3.p2.p1.g0) + (p8.p7.p6.p5.p4.p3.p2.p1.p0.c0)
c9 = g9 + (p9.g8) + (p9.p8.g7) + (p9.p8.p7.g6) + (p9.p8.p7.p6.g5) + (p9.p8.p7.p6.p5.g4) + (p9.p8.p7.p6.p5.p4.g3)+ (p9.p8.p7.p6.p5.p4.p3.g2) + (p9.p8.p7.p6.p5.p4.p3.p2.g1) + (p9.p8.p7.p6.p5.p4.p3.p2.p1.g0) + (p9.p8.p7.p6.p5.p4.p3.p2.p1.p0.c0)

Required Reading : Section B.6 until (and including) section
“Fast Carry Using the First Level of Abstraction: Propagate and Generate”

Getting Started :

This lab requires to use a terminal on a linux OS. Use vole.cselabse.umn.edu to login to a virtual linux desktop, or alternately, use one of the linux CSE Labs machines in Keller Hall. Unzip the handin folder using the command :

```
tar -xvf  Lab1.tar.gz
```

A folder `Lab1` will be created with base files in subfolder `Base` and testbenches in `Testbenches` subfolder.

You need to modify the following files for Problem 1 :

```
one_bit_ALU.v ripple_carry_ten_bit.v
```

You need to modify the following files for Problem 2 :

```
Carry_Block_1.v Carry_Block_2.v Carry_Block_3.v Carry_Block_4.v
Carry_Block_5.v Carry_Block_6.v Carry_Block_7.v Carry_Block_8.v
Carry_Lookahead_Adder.v PG_Block.v
```

You may create a new Vivado project for each problem and add these files to the project.

Grader :
Once you have completed your solution to both problems, you can grade it.
The provided automated grader exhaustively tests your solution for the correctness.
Copy the below files of your implementation into the subfolder `Grader`.

```
one_bit_ALU.v ripple_carry_ten_bit.v
Carry_Block_1.v Carry_Block_2.v Carry_Block_3.v Carry_Block_4.v
Carry_Block_5.v Carry_Block_6.v Carry_Block_7.v Carry_Block_8.v
Carry_Lookahead_Adder.v PG_Block.v
```

Also copy the following files from the `Testbenches` folder into the `Grader` folder :

```
tb_one_bit_ALU.v tb_ripple_carry_10_bit.v tb_Carry_Lookahead_Adder.v
```

Then, navigate into the `Grader` folder and run the grader using the command :

```
./grade.sh
```

The grader will print how many test cases failed and your total score, for each problem.

Submission instructions :

Once you have your final versions of the files, create a folder Lab1_<name> (where name is your first name). Copy the above verilog files (exactly the ones listed in the grader section) to the folder Lab1_<name>. Please do not submit any testbenches (the files named as tb_*.v). Please do not create any subfolders. Please do not submit any other files such as temp files created by Vivado. Create a .tar.gz file using the command →

```
tar -czvf Lab1_<name>.tar.gz Lab1_<name>
```

For example, I would use the command :

```
tar -czvf Lab1_Kartik.tar.gz Lab1_Kartik
```

Finally, submit this on the moodle link for ‘Lab Assignment 1’.

Note :
Follow the naming convention for your submission in order to facilitate automatic grading
Eg: The folder you submit should be Lab1_<name>, not lab1_<name> or _Lab1_<name> or Lab1.<name> or other variations. The tar file you submit should be of format Lab1_<name>.tar.gz, not Lab1_<name>.tar, or Lab1_<name>.zip or other variations.